AD-A060 388 SOUTHERN METHODIST UNIV DALLAS TX DEPT OF OPERATIONS--ETC F/G 12/1
SQUEEZE METHODS FOR GENERATING GAMMA VARIATES.(U)
AUG 78 B W SCHMEISER N00014-77-C-0425
UNCLASSIFIED OREM-78009 NL

1 OF 1
AD
A0 60 388

END
DATE
FILMED
1-79
DDC

12

AD A060388

9 Technical Report OREM-78009 14

6 SQUEEZE METHODS FOR GENERATING GAMMA VARIATES.

10 Bruce W. Schmeiser

Department of Operations Research
and Engineering Management
Southern Methodist University
Dallas, Texas 75275

12 24p.

July 1978

Revised August 1978 11

This work supported by the Office of Naval Research under Contract N00014-77-C-0425 15

ABSTRACT

Two algorithms are given for generating gamma distributed random variables. The algorithms, which are valid when the shape parameter is greater than one, use a uniform majorizing function for the body of the distribution and exponential majorizing functions for the tails. The algorithms are self-contained, requiring only U(0,1) variates. Comparisons are made to three competitive algorithms in terms of marginal generation times, initialization time, and memory requirements. Both algorithms are faster than existing methods, for all values of the shape parameter.

KEY WORDS

Gamma Distribution

Simulation

Random Number Generation

Rejection Methods

Monte Carlo

Distribution Sampling

1. INTRODUCTION

During the last few years many algorithms have been developed for generation of gamma random variables having density function

$$f_\gamma(x) = x^{\alpha-1} \exp(-x)/\Gamma(\alpha) \qquad 0 \leq x < \infty,\ 1 < \alpha < \infty.$$

To the author's knowledge, these include Ahrens and Dieter (1974), Atkinson and Pearce (1976), Fishman (1973), Fishman (1976), Jöhnk (1974), Greenwood (1974), Marsaglia (1977), McGrath and Irving (1973), Tadikamalla (1978a, 1978b), C.S. Wallace (1976), N.D. Wallace (1974), and Whittaker (1974). Most have been implementations of the general acceptance/rejection algorithm, with many using the modification referred to as the "squeeze" technique by Marsaglia (1977). The algorithms developed in this paper use the squeeze technique, which in the general case proceeds as follows:

Let $f(x)$ be the density function from which random variates are desired and let $t(x)$ and $b(x)$ be majorizing and minorizing functions of $f(x)$, respectively ($t(x) \geq f(x)$ for all x and $b(x) \leq f(x)$ for all x). Then

1. Generate x having density $r(x) = t(x)/\int_{-\infty}^{\infty} t(y)dy$.
2. Generate $v \sim U(0,1)$.
3. If $v \leq b(x)/t(x)$, deliver x.
4. If $v \leq f(x)/t(x)$, deliver x. Otherwise go to step 1.

If $t(x)$ fits $f(x)$ well, if $r(x)$ yields variates quickly, and if $b(x)$ both fits $f(x)$ well and is quick to evaluate, the squeeze technique yields variates quickly even when $f(x)$ is time consuming to evaluate.

2. The Algorithms

Similar to the beta algorithms of Schmeiser and Shalaby (1977), the points of inflection and the mode are central to this algorithm. Define

$$x_1 = x_2(1 - 1/(x_3 - x_2))$$

$$x_2 = \text{Max}(0,\ x_3 - x_3^{1/2})$$

$$x_3 = \alpha - 1$$

$$x_4 = x_3 + x_3^{1/2}$$

$$x_5 = x_4(1 + 1/(x_4 - x_3))$$

Here x_3 is the mode, x_2 and x_4 are the points of inflection of $f(x)$ and x_1 and x_5 are the points at which the tangent of $f(x)$ at x_2 and x_4 cross the X axis. If $\alpha < 2$, there is no left point of inflection and $x_1 = x_2 = 0$. These points are illustrated in Figure A.

Figure A About Here

The simpler, and slower, of the two algorithms is described first. The algorithm uses a uniform majorizing function for the body of the distribution and an exponential majorizing function for the tails. It is denoted G2PE since it is a gamma (G) generator which requires evaluating the density function at two points (2P) and uses an exponential (E) majorizing function for the tails.

For simplicity $f_\gamma(x)$ is rescaled to

$$f(x) = \exp[x_3\ \ell n(x/x_3) + x_3 - x]$$

to avoid evaluating the gamma function and to yield $f(x_3) = 1$.

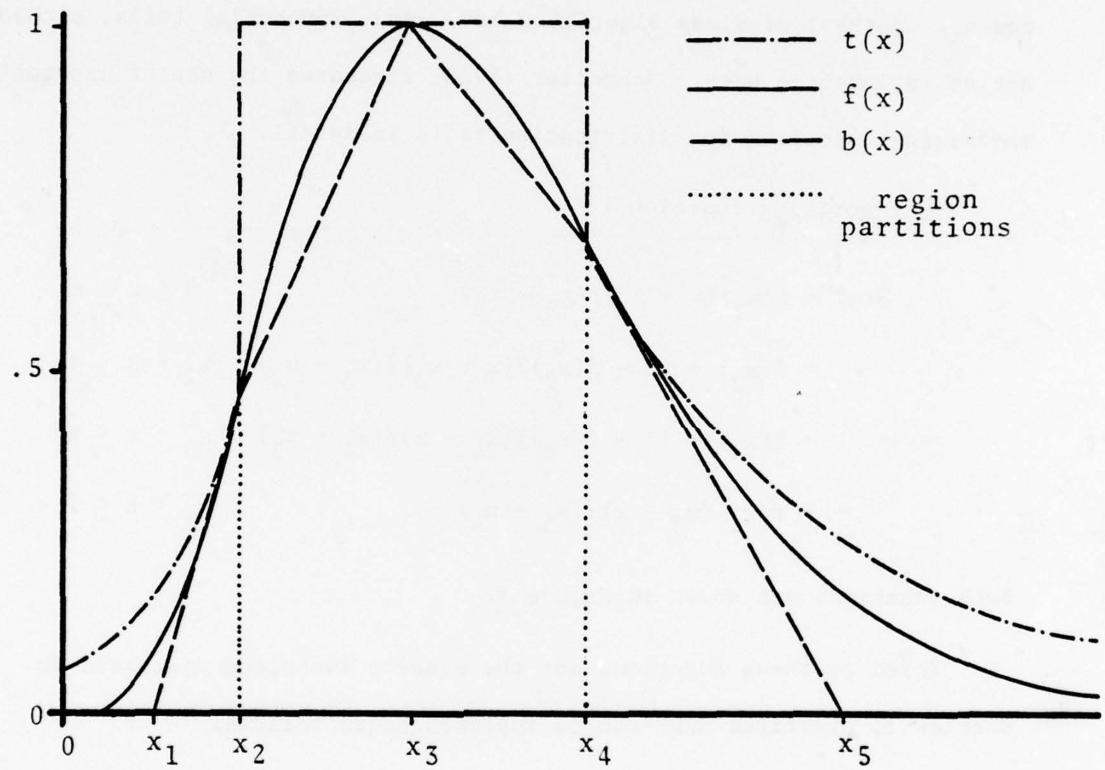

Figure A. Algorithm G2PE for α = 5.

The majorizing function is

$$
\begin{aligned}
t(x) &= f(x_2)\ \exp(-\lambda_L(x - x_2)) && 0 < x \le x_2 \\
&= 1 && x_2 < x \le x_4 \\
&= f(x_4)\exp(-\lambda_R(x - x_4)) && x_4 < x < \infty
\end{aligned}
$$

where $\lambda_L = 1-(x_3/x_2)$ and $\lambda_R = 1-(x_3/x_4)$ to make $t(x)$ tangent to $f(x)$ at x_2 and x_4. Several previous algorithms have used exponential tails, although not as implemented here. Schmeiser (1978) discusses the use of exponential majorizing functions for distribution tails in detail.

The minorizing function is

$$
\begin{aligned}
b(x) &= f(x_2)(x - x_1)/(x_2 - x_1) && 0 < x \le x_2 \\
&= f(x_2) + (1 - f(x_2))(x - x_2)/(x_3 - x_2) && x_2 < x \le x_3 \\
&= f(x_4) + (1 - f(x_4))(x_4 - x)/(x_4 - x_3) && x_3 < x \le x_4 \\
&= f(x_4)(x_5 - x)/(x_5 - x_4) && x_4 < x \le 1
\end{aligned}
$$

Both functions are shown in Figure A.

Based on these functions and the squeeze technique discussed in Section 1, algorithm G2PE can be implemented as follows.

<u>Algorithm G2PE</u>

Initialization

1. Set $x_3 = \alpha - 1$, $D = x_3^{1/2}$, $\lambda_L = 1$, $x_1 = x_2 = f_2 = 0$.
   If $D \ge x_3$ go to step 2. Otherwise set
   $x_2 = x_3 - D$, $\lambda_L = 1 - x_3/x_2$, $x_1 = x_2 + 1/\lambda_L$,
   and $f_2 = f(x_2)$.

2. Set $x_4 = x_3 + D$, $\lambda_R = 1 - x_3/x_4$, $x_5 = x_4 + 1/\lambda_R$.

$f_4 = f(x_4)$, $p_1 = x_4 - x_2$, $p_2 = p_1 - f_2/\lambda_L$,

$p_3 = p_2 + f_4/\lambda_R$.

Generation

3. Sample $u, v \sim U(0,1)$ and set $u = up_3$.

If $u > p_1$, go to step 4. Otherwise set $x = x_2 + u$. If $x > x_3$ and $v \leq f_4 + (x_4 - x)(1 - f_4)/(x_4 - x_3))$, deliver x. If $x < x_3$ and $v \leq f_2 + (x - x_2)(1 - f_2)/(x_3 - x_2)$, deliver x. Otherwise go to step 6.

4. If $u > p_2$, go to step 5. Otherwise set $u = (u - p_1)/(p_2 - p_1)$, $x = x_2 - \ln(u)/\lambda_L$. If $x < 0$, go to step 3. Otherwise set $v = vf_2u$. If $v \leq f_2(x - x_1)/(x_2 - x_1)$, deliver x. Otherwise go to step 6.

5. Set $u = (u - p_2)/(p_3 - p_2)$, $x = x_4 - \ln(u)/\lambda_R$, $v = vf_4u$. If $v \leq f_4(x_5 - x)/(x_5 - x_4)$, deliver x.

6. If $\ln v \leq x_3 \ln(x/x_3) + x_3 - x$, deliver x. Otherwise go to step 3.

The second algorithm, denoted G4PE for reasons analogous to G2PE, is illustrated in Figure B. The majorizing function is uniform over the body of the distribution, triangular over the shoulders, and exponential in the tails. The resulting area under the majorizing function is partitioned into the ten regions shown. Four regions have zero probability of rejection. Of the remaining six regions, two require uniform variates, two require triangular variates and two require exponential variates.

The algorithm may be implemented as follows:

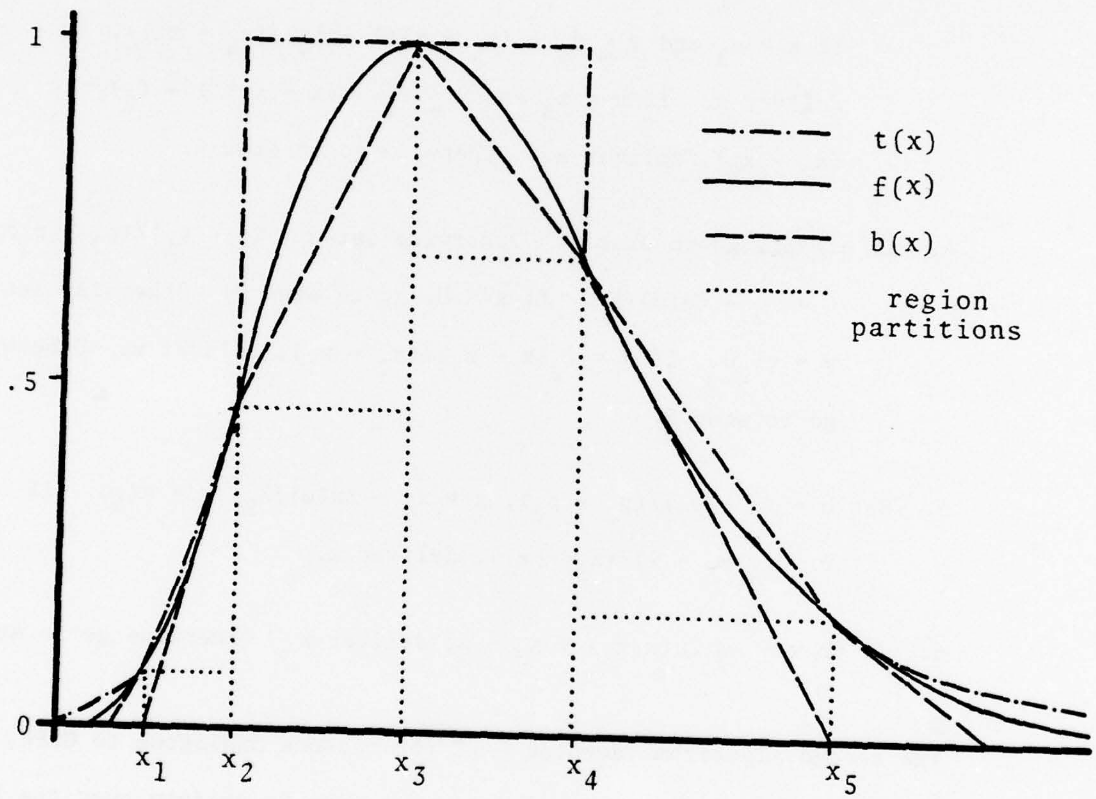

Figure B. Algorithm G4PE for α = 5.

Algorithm G4PE

Initialization

1. Set $x_3 = \alpha - 1$, $D = x_3^{1/2}$, $x_1 = x_2 = f_1 = f_2 = 0$.
   If $D \geq x_3$, go to step 2. Otherwise set $x_2 = x_3 - D$,
   $x_1 = x_2(1 - 1/D)$, $\lambda_L = 1 - x_3/x_1$, $f_1 = f(x_1)$, and $f_2 = f(x_2)$.

2. Set $x_4 = x_3 + D$, $x_5 = x_4(1 + 1/D)$, $\lambda_R = 1 - x_3/x_5$,
   $f_4 = f(x_4)$, and $f_5 = f(x_5)$. Set $p_1 = f_2(x_3 - x_2)$,
   $p_2 = f_4(x_4 - x_3) + p_1$, $p_3 = f_1(x_2 - x_1) + p_2$,
   $p_4 = f_5(x_5 - x_4) + p_3$, $p_5 = (1 - f_2)(x_3 - x_2) + p_4$,
   $p_6 = (1 - f_4)(x_4 - x_3) + p_5$, $p_7 = (f_2 - f_1)(x_2 - x_1)/2 + p_6$,
   $p_8 = (f_4 - f_5)(x_5 - x_4)/2 + p_7$, $p_9 = -f_1/\lambda_L + p_8$,
   $p_{10} = f_5/\lambda_R + p_9$.

Generation

3. Sample $u \sim U(0,1)$ and set $u = u\, p_{10}$. If $u > p_4$, go to step 7.
   If $u > p_1$, go to step 4. Otherwise deliver $x = x_2 + u/f_2$.

4. If $u > p_2$, go to step 5. Otherwise deliver $x = x_3 + (u - p_1)/f_4$.

5. If $u > p_3$, go to step 6. Otherwise deliver $x = x_1 + (u - p_2)/f_1$.

6. Deliver $x = x_4 + (u - p_3)/f_5$.

7. Sample $w \sim U(0,1)$. If $u > p_5$, go to step 8. Otherwise
   set $x = x_2 + (x_3 - x_2)w$. If $(u - p_4)/(p_5 - p_4) \leq w$, deliver x.
   Otherwise set $v = f_2 + (u - p_4)/(x_3 - x_2)$ and go to step 13.

8. If $u > p_6$, go to step 9. Otherwise set $x = x_3 + (x_4 - x_3)w$.
   If $(p_6 - u)/(p_6 - p_5) \geq w$, deliver x. Otherwise set
   $v = f_4 + (u - p_5)/(x_4 - x_3)$ and go to step 13.

9. If $u > p_8$, go to step 11. Otherwise sample $w_2 \sim U(0,1)$. If $w_2 > w$, set $w = w_2$. If $u > p_7$, go to step 10. Otherwise set $x = x_1 + (x_2 - x_1)w$, $v = f_1 + 2w(u - p_6)/(x_2 - x_1)$. If $v \leq f_2 w$, deliver x. Otherwise go to step 13.

10. Set $x = x_5 - w(x_5 - x_4)$, $v = f_5 + 2w(u - p_7)/(x_5 - x_4)$ and go to step 13.

11. If $u > p_9$, go to step 12. Otherwise set $u = (p_9 - u)/(p_9 - p_8)$, $x = x_1 - (\ln u)/\lambda_L$. If $x \leq 0$, go to step 3. If $w < (\lambda_L(x_1 - x) + 1)/u$, deliver x. Otherwise set $v = w\, f_1\, u$ and go to step 13.

12. Set $u = (p_{10} - u)/(p_{10} - p_9)$, $x = x_5 - (\ln u)/\lambda_R$. If $w < (\lambda_R(x_5 - x) + 1)/u$, deliver x. Otherwise set $v = w\, f_5\, u$.

13. If $\ln v > f(x)$, go to step 3. Otherwise deliver x.

3. COMPUTATIONAL RESULTS

Based on the findings of Cheng (1976), Marsaglia (1977) and Tadikamalla (1978b), it appears that the three fastest existing algorithms are to be found in these three papers. Using the names used in the above papers, Cheng's algorithm is denoted by GB, Marsaglia's algorithm is denoted RGAMA, and Tadikamalla's algorithm is denoted by GAMMA.

The table compares G2PE, G4PE, GB, RGAMA and GAMMA in terms of generation time per variate, initialization time, and memory requirements. The times are based on the generation of 10,000 variates and are accurate to within .02 milliseconds. The algorithms were coded in FORTRAN on the CDC Cyber 72 at Southern Methodist University. The uniform variates were generated by the relatively fast RANF internal to the FTN compiler.

Insert Table About Here

It is clear that all five algorithms are robust to changes in α, with all algorithms but GAMMA being slightly faster for larger α. In this implementation, the algorithms can be ranked in order of increasing marginal times as G4PE, G2PE, RGAMA, GB and GAMMA, except that GAMMA is faster than GB for $\alpha < 1.5$. G2PE is about 15% faster, and G4PE is 30-40% faster, than the previous fastest algorithm RGAMA.

Marginal Generation Times (in Milliseconds)

and Memory Requirements

	Method				
α	RGAMA[a]	GAMMA	GB	G2PE	G4PE
1.0001	.53	.56	.70	.46	.39
1.2	.53	.61	.67	.45	.33
1.5	.52	.64	.63	.41	.33
2	.52	.70	.62	.42	.33
3	.49	.71	.60	.40	.29
5	.49	.75	.56	.37	.28
8	.47	.76	.57	.39	.28
20	.46	.78	.54	.40	.28
100	.47	.76	.55	.38	.26
1000	.45	.72	.53	.38	.26
Set-up Time	.24	.34	.18	.53-.83[b]	1.01-1.57[b]
Memory Requirements[c]	538	316	290	405	566

[a]As implemented using the KR normal generator. For the implementation using the polar method, add approximately .09 milliseconds for all α and decrease the memory requirements by 224.

[b]Depends upon the value of α. The lower set-up time applies when $\alpha \leq 2$ and the higher value corresponds to $\alpha > 2$.

[c]Memory requirements include necessary routines such as ALOG, EXP and SQRT.

Each of the algorithms requires a one time initialization. In order of increasing set-up time the algorithms are GB, RGAMA, GAMMA, G2PE and G4PE. Since the algorithms with lower marginal times tend to have higher set-up times, a tradeoff is made which depends upon M, the required number of variates for a fixed α. For $\alpha \leq 2$, RGAMA is fastest for $M \leq 3$, G2PE is fastest for M = 4 or 5, and G4PE is fastest for $M \geq 6$. For $\alpha > 2$, RGAMA is fastest for $M < 7$ and G4PE is fastest for $M > 7$. For no combination of α and M is either GAMMA or GB the fastest algorithm.

Memory requirements are also shown in the table. In order of increasing memory requirements the algorithms are GB, GAMMA, G2PE, RGAMA, and G4PE which includes necessary routines such as ALOG, EXP and SQRT. However RGAMA requires a normal variate generator, which as implemented here is algorithm KR given by Kinderman and Ramage (1976) with memory requirements of 289. While a normal variate algorithm requiring less memory could be used, marginal generation times would increase for RGAMA. For example, using Marsaglia's polar method, total memory requirements for RGAMA were only 314, but marginal generation times increased approximately .09 millisecond for all values of α. Of course different normal generators will result in various tradeoffs between speed and memory.

Ease of implementation may be crudely measured in lines of code and additional algorithms needed. In ascending order of lines of code the algorithms are GB, RGAMA, GAMMA, G2PE and G4PE. The only additional algorithm needed is the normal generator used by RGAMA.

REFERENCES

Ahrens, J. H. and Dieter, U. (1974), "Computer Methods for Sampling From Gamma, Beta, Poisson and Binomial Distributions," Computing, 12, 223-246.

Atkinson, A. C., and Pearce, M. C. (1976), "The Computer Generation of Beta, Gamma and Normal Random Variables," Journal of the Royal Statistical Society, A, 139, 431-461.

Cheng, R. C. H. (1977), "The Generation of Gamma Variables with Non-integer Shape Parameter," Applied Statistics, 26, 71-75.

Fishman, G. S. (1973), Concepts and Methods in Discrete Event Digital Simulation, New York: John Wiley & Sons.

Fishman, G. S. (1976), "Sampling from the Gamma Distribution on a Computer," Communications of the ACM, 19, 407-409.

Forsythe, G. E. (1972), "Von Neumann's Comparison Method for Random Sampling from the Normal and Other Distributions," Mathematics of Computation, 26, 817-826.

Greenwood, A. J. (1974), "A Fast Generator for Gamma-distributed Random Variables," COMPSTAT: Proceedings in Computational Statistics (G. Bruckman, F. Ferschl, L. Schmetterer, eds), 17-27, Physica-Verlag, Vienna.

Jöhnk, M. D. (1964), "Erzeugung von Betaverteilten und Gammaverteilten Zufallszahlen," Metrika, 8, 5-15.

Kinderman, A. J., and Ramage, J. G. (1976), "Computer Generation of Normal Random Variables," Journal of the American Statistical Association, 71, 893-896.

Marsaglia, George (1977), "The Squeeze Method for Generating Gamma Variates," Computers and Mathematics with Applications, 3, 321-325.

McGrath, E. J., and Irving, D. C. (1973), Techniques for Efficient Monte Carlo Simulation. Vol. II. Random Number Generation for Selected Probability Distributions, Springfield, Virginia: National Technical Information Service.

Schmeiser, B. W., and Shalaby, M. A. (1977), "Rejection Methods for Beta Variate Generation," Technical Report 77014, Department of Operations Research and Engineering Management, Southern Methodist University, Dallas, TX 75275.

Schmeiser, B. W. (1978), "Generation of Variates from Distribution Tails," Technical Report 78008, Department of Operations Research and Engineering Management, Southern Methodist University, Dallas, TX 75275.

Tadikamalla, P. R. (1978a), "Computer Generation of Gamma Random Variables," Communications of the ACM, 21, 419-422.

Tadikamalla, P. R. (1978b), "Computer Generation of Gamma Random Variables-II," Communications of the ACM, forthcoming.

Wallace, C. S. (1976), "Transformed Rejection Generators for Gamma and Normal Pseudo-Random Variables," The Australian Computer Journal, 8, 103-109.

Wallace, N. D. (1974), "Computer Generation of Gamma Random Variates with Non-integral Shape Parameters," Communications of the ACM, 17, 691-695.

Whittaker, J. (1974), "Generating Gamma and Beta Random Variables with Non-integral Shape Parameters," Applied Statistics, 23, 210-214.

APPENDIX

```
      PROGRAM MAIN (INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)
C     BRUCE SCHMEISER  JULY 12,1978  SOUTHERN METHODIST UNIVERSITY
C     TO COMPARE GAMMA VARIATE GENERATION ROUTINES
C     A = SHAPE PARAMETER OF THE GAMMA DISTRIBUTION
      DIMENSION NAME(5),AS(10)
      DATA NAME/≠TADI≠,≠MARS≠,≠CHNG≠,≠G2PE≠,≠G4PE≠/
      DATA AS/1.0001,1.2,1.5,2.,3.,5.,8.,20.,100.,1000./
      BETA = 1.
      N = 10000
C
      DO 10 I=1,10
      ALPHA = AS(I)
C
      DO 20 J=1,5
      SUM = 0.
      SUM2 = 0.
      TIME = SECOND(X)
C
      DO 30 K=1,N
      GO TO (1,2,3,4,5), J
    1 CALL GAMMA(ALPHA,BETA,ISEED,X)
      GO TO 15
    2 CALL RGAM(ALPHA,BETA,ISEED,X)
      GO TO 15
    3 CALL GR(ALPHA,BETA,ISEED,X)
      GO TO 15
    4 CALL G2PE(ALPHA,BETA,ISEED,X)
      GO TO 15
    5 CALL G4PE(ALPHA,BETA,ISEED,X)
   15 SUM = SUM + X
   30 SUM2 = SUM2 + X*X
C
      TIME = (SECOND(X)-TIME)*1000/N
      SUM = SUM / N
      SUM2 = SUM2/N - SUM*SUM
   20 WRITE (6,101) NAME(J),ALPHA,SUM,SUM2,TIME,N
  101 FORMAT (≠0≠,A4,4F10.4,I6)
   10 CONTINUE
      STOP
      END
```

```
      SUBROUTINE RGAM (A,BETA,ISEED,RGAMA)
C     BRUCE SCHMEISER   JULY 12,1978   SOUTHERN METHODIST UNIVERSITY
C       TO GENERATE STANDARD GAMMA VARIATES USING MARSAGLIA*S SQUEEZE METHOD
C       REFERENCE HIS ARTICLE IN COMP. AND MATH. WITH APPLICATIONS
C       VOL 3,PP.321-325.  1977
C       A = SHAPE PARAMETER
C       X = GENERATED VARIATE
C       A MUST BE GREATER THAN 1/3 AND HE RECOMMENDS A .GT. 1
C
      DATA B/1./
      IF (B .EQ. A)  GO TO 1
      B = A
      S = .3333333/SQRT(A)
      ZO = 1.-1.732051*S
      CC = A * ZO**3 -.5*(S-1.732051)**2
      CL = 3.*A-1.
      CS = 1.-S*S
C
C     REJECTION PROCEDURE BEGINS HERE
C
    1 CALL NORMAL(ISEED,X)
      Z = S*X + CS
      IF (Z .LE. O.)  GO TO 1
      RGAMA = A*Z**3
      E = -ALOG(RANF(ISEED))
      CD = E + .5*X**2 - RGAMA + CC
      T = 1. - ZO/Z
      IF (CD + CL*T*(1.+T*(.5+.3333333*T)) .GT. O.)  GO TO 2
      IF (CD + CL*ALOG(Z/ZO) .LT. O.)  GO TO 1
    2 RGAMA = RGAMA*BETA
      RETURN
      END



      SUBROUTINE NORMAL(ISEED,X)
C
C         GENERATION OF ONE NORMAL(0,1) VARIATE USING
C         THE ALGORITHM GIVEN BY KINDERMAN AND RAMAGE
C         IN THE JOURNAL OF THE AMERICAN STATISTICAL ASSOCIATION 12/76
C
C         CODED BY PETER BONNER AND MODIFIED BY BRUCE SCHMEISER
C         MARCH 1977 AND JUNE 1977 RESPECTIVELY
C
C
      DATA TAIL/2.216035867166471/
      UU=RANF(ISEED)
      IF(UU.GE..884070402298758) GO TO 2
C
C         RETURN TRIANGULAR VARIATE 88 PERCENT OF THE TIME
C
      Y=RANF(ISEED)
      X=TAIL*(1.131131635444180*UU+Y-1.0)
```

```
      RETURN
    2 IF(UU.LT..973310954173898) GO TO 4
C
C         TAIL COMPUTATION
C
    3 V=RANF(ISEED)
      W=RANF(ISEED)
      T1=TAIL*TAIL/2.0
      T=T1-ALOG(W)
      IF(V*V*T.GT.T1) GO TO 3
      X=SQRT(2.0*T)
      IF(UU.GE..986655477086949) X=-X
      RETURN
    4 IF(UU.LT..958720824790463) GO TO 6
C
C         FIRST NEARLY LINEAR DENSITY
C
    5 V=RANF(ISEED)
      W=RANF(ISEED)
      Z=V-W
C     LET V= MAX(V,W) AND LET W=MIN(V,W)
      IF(V.GT.W) GO TO 100
      TEMP=V
      V=W
      W=TEMP
  100 T=TAIL-.630834801921960*W
      IF(V.LE..755591531667601) GO TO 9
      DIFF=EXP(-T*T*.5)/2.50662827463100-.180025191068563*
     *     (2.216035867166471-ABS(T))
      IF(ABS(Z)*.034240503750111.LE.DIFF)  GO TO 9
      GO TO 5
    6 IF(UU.LT..911312780288703) GO TO 8
C
C         SECOND NEARLY LINEAR DENSITY
C
    7 V=RANF(ISEED)
      W=RANF(ISEED)
      Z=V-W
C     LET V= MAX(V,W) AND LET W=MIN(V,W)
      IF(V.GT.W) GO TO 101
      TEMP=V
      V=W
      W=TEMP
  101 T=.479727404222441+1.105473661022070*W
      IF(V.LE..872834976671790) GO TO 9
      DIFF=EXP(-T*T*.5)/2.50662827463100-.180025191068563*
     *     (2.216035867166471-ABS(T))
      IF(ABS(Z)*.049264496373128.LE.DIFF)  GO TO 9
      GO TO 7
C
C         THIRD NEARLY LINEAR DENSITY
C
    8 V=RANF(ISEED)
      W=RANF(ISEED)
```

```
      Z=V-W
C     LET V= MAX(V,W) AND LET W=MIN(V,W)
      IF(V.GT.W) GO TO 102
      TEMP=V
      V=W
      W=TEMP
  102 T=.479727404222441-.595507138015940*W
      IF(V.LE..805577924423817) GO TO 9
      DIFF=EXP(-T*T*.5)/2.506628274631000-.180025191068563*
     *     (2.216035867166471-ABS(T))
      IF(ABS(Z)*.053377549506886.LE.DIFF)  GO TO 9
      GO TO 8
    9 X=T
      IF(Z.GE.0.0) X=-X
      RETURN
      END




      SUBROUTINE GB (ALPHA,BETA,ISEED,X)
C     BRUCE SCHMEISER   AUGUST 4,1978  SOUTHERN METHODIST UNIVER
C     GAMMA VARIATE GENERATOR.  REFERENCE CHENG, APPLIED STATIS
C       (1977),26,1,71-75.
C     ALPHA = SHAPE PARAMETER
C     BETA = SCALE PARAMETER
C     ISEED = RANDOM NUMBER SEED
C     X = GENERATED GAMMA VARIATE
      DATA ASAVE /-1./
      IF (ALPHA .EQ. ASAVE)  GO TO 100
C
C*****INITIALIZATION
C
      ASAVE = ALPHA
      A = 1. / SQRT(ALPHA+ALPHA - 1.)
      B = ALPHA - 1.38629
      C = ALPHA + 1./A
C
C*****GENERATION OF ONE GAMMA VARIATE X
C
  100 U1 = RANF(ISEED)
      U2 = RANF(ISEED)
      V = A * ALOG(U1/(1.-U1))
      X = ALPHA * EXP(V)
      Z = U1*U1*U2
      R = B + C*V - X
      IF (R + 2.5040774 - 4.5*Z .GE. 0.)  GO TO 200
      IF (R .LT. ALOG(Z))  GO TO 100
  200 X = BETA*X
      RETURN
      END
```

```
      SUBROUTINE G2PE(ALPHA,BETA,ISEED,X)
C*****BRUCE SCHMEISER   JULY 12,1978   SOUTHERN METHODIST UNIVERSITY
C     TO GENERATE A STANDARD GAMMA VARIATE USING EXPONENTIAL TAIL
C       REJECTION AND RECTANGULAR REJECTION  FOR THE BODY OF THE
C       DISTRIBUTION.
C       A = THE SHAPE PARAMETER (A .GT. 1)
C       X = THE GENERATED VALUE
C
      DATA ASAVE/-1./,XLL/1./
      IF (ALPHA .EQ. ASAVE)  GO TO 100
C
C***** SET-UP BEGINS HERE
C
      ASAVE = ALPHA
      X1 = 0.
      X2 = 0.
      F2= 0.
      X3 = ALPHA - 1
      D = SQRT(X3)
      IF (D .GE. X3)  GO TO 10
      X2 = X3-D
      XLL = 1. - X3/X2
      X1 = X2 + 1/XLL
      F2 = EXP(X3*ALOG(X2/X3) + X3 - X2)
   10 X4 = X3 + D
      XLR = 1. - X3/X4
      X5 = X4 + 1/XLR
      F4 = EXP(X3*ALOG(X4/X3) + X3 - X4)
      P1 = X4-X2
      P2 = P1 - F2/XLL
      P3 = P2 + F4/XLR
C
C*****VARIATE GENERATION PROCEDURE BEGINS HERE
C
  100 U = RANF(ISEED)*P3
      V = RANF(ISEED)
C
C     RECTANGULAR REJECTION
C
      IF (U .GT. P1)  GO TO 200
      X = X2 + U
      IF (X .LT. X3)  GO TO 110
      IF (V .LT. F4 + (X4-X)*(1-F4)/(X4-X3))  GO TO 500
      GO TO 400
  110 IF (V .LT. F2 + (X-X2)*(1-F2)/(X3-X2))  GO TO 500
      GO TO 400
C
C     LEFT TAIL GENERATION
C
  200 IF (U .GT. P2)  GO TO 300
      U = (U-P1)/(P2-P1)
      X = X2 - ALOG(U) / XLL
      IF (X .LT. 0.)  GO TO 100
      V = V * F2 * U
      IF (V .LT. F2 * (X-X1)/(X2-X1))  GO TO 500
      GO TO 400
```

```
C
C     RIGHT TAIL GENERATION
C
  300 U = (U-P2) / (P3-P2)
      X = X4 - ALOG(U) / XLR
      V = V * F4 * U
      IF (V .LT. F4*(X5-X)/(X5-X4))  GO TO 500
C
C     FINAL REJECTION TRY
C
  400 IF(ALOG(V) .GT. X3*ALOG(X/X3) + X3 - X)  GO TO 100
  500 X = X*BETA
      RETURN
      END



      SUBROUTINE GAMMA (ALPHA, BETA, ISEED, X)
C
C     BRUCE SCHMEISER SOUTHERN METHODIST UNIVERSITY  MAY 24, 1978
C     TO GENERATE GAMMA VARIATES WITH MEAN ALPHA*BETA
C       AND VARIANCE ALPHA*BETA*BETA USING TADIKAMALLA*S ALGORITHM
C       DESCRIBED IN GENERATION OF GAMMA VARIATES -- II
C       TO APPEAR IN CACM.
C     VALID ONLY FOR ALPHA .GT. 1
C
      DATA ASAVE/-1./
      IF (ALPHA .EQ. ASAVE)  GO TO 100
C
C       SET - UP CONSTANTS
C
      ASAVE = ALPHA
      A = ALPHA - 1.
      B = .5 + .5*SQRT (4.*ALPHA-3)
      C = A * (1.+B) / B
      D = (B-1.) / (A*B)
      E = EXP(-A/B) / 2.
C
C       GENERATION OF ONE VARIATE
C
  100 U = E + RANF(ISEED) * (1.-E)
      IF (U .GT. .5)  GO TO 200
      X = A + B*ALOG(U+U)
      IF (X .LT. 0.)  GO TO 100
      Y = A - X
      GO TO 300
  200 X = A - B*ALOG(2.-U-U)
      Y = X - A
  300 U = RANF(ISEED)
      IF (ALOG(U) .GT. (A*ALOG(D*X)-X+(Y/B)+C))  GO TO 100
      X = X*BETA
      RETURN
      END
```

```
          SUBROUTINE G4PE (ALPHA,BETA,ISEED,X)
C         BRUCE SCHMEISER   JULY 1978   SOUTHERN METHODIST UNIVERSIT
C         GENERATION OF ONE PSEUDO-RANDOM VARIATE USING
C         THE FOUR POINT METHOD WITH EXPONENTIAL TAILS
C         FROM THE GAMMA DENSITY FUNCTION
C         ALPHA = SHAPE PARAMETER
C         BETA = SCALE PARAMETER
C         ISEED = RANDOM NUMBER SEED
C         X = GENERATED GAMMA VARIATE
C         REFERENCE  B. SCHMEISER  SQEEZE METHODS FOR GAMMA VARIAT
C           GENERATION, OREM TECH REPORT 78009  JULY 1978
          DATA ASAVE /-1./,XLL/-1./
          IF (ALPHA .EQ. ASAVE)  GO TO 100
C
C*****INITIALIZATION
C
          ASAVE = ALPHA
1.        X1 = X2 = F1 = F2 = 0.
          X3 = ALPHA - 1.
          D = SQRT(X3)
          IF (D .GE. X3)  GO TO 10
          X2 = X3 - D
          X1 = X2*(1.-1./D)
          XLL = 1.-X3/X1
          F1 = EXP (X3*ALOG(X1/X3) + X3 - X1)
          F2 = EXP (X3*ALOG(X2/X3) + X3 - X2)
2.     10 X4 = X3 + D
          X5 = X4*(1.+1./D)
          XLR = 1. - X3/X5
          F4 = EXP (X3*ALOG(X4/X3) + X3 - X4)
          F5 = EXP (X3*ALOG(X5/X3) + X3 - X5)
C
C         CALCULATE PROBABILITY FOR EACH OF THE TEN REGIONS
C
          P1 = F2*(X3-X2)
          P2 = F4*(X4-X3) + P1
          P3 = F1*(X2-X1) + P2
          P4 = F5*(X5-X4) + P3
          P5 = (1.-F2)*(X3-X2) + P4
          P6 = (1.-F4)*(X4-X3) + P5
          P7 = (F2-F1)*(X2-X1)*.5 + P6
          P8 = (F4-F5)*(X5-X4)*.5 + P7
          P9 = -F1/XLL + P8
          P10 = F5/XLR + P9
C
C*****GENERATE ONE GAMMA VARIATE X
C
3.    100 U = RANF(ISEED) * P10
C
C            THE FOUR REGIONS WITH ZERO PROBABILITY OF REJECTION
C
          IF (U .GT. P4)  GO TO 500
          IF (U .GT. P1)  GO TO 200
          X = X2 + U/F2
          GO TO 1400
4.    200 IF (U .GT. P2)  GO TO 300
          X = X3 + (U-P1)/F4
```

```
         GO TO 1400
5.   300 IF (U .GT. F3)  GO TO 400
         X = X1 + (U-F2) / F1
         GO TO 1400
6.   400 X = X4 + (U-F3) / F5
         GO TO 1400
C
C        THE TWO REGIONS USING RECTANGULAR REJECTION
C
7.   500 W = RANF(ISEED)
         IF (U .GT. P5)  GO TO 600
         X = X2 + (X3-X2) * W
         IF ((U-P4)/(F5-P4) .LE. W)  GO TO 1400
         V = F2 + (U-F4) / (X3-X2)
         GO TO 1300
8.   600 IF (U .GT. P6)  GO TO 700
         X = X3 + (X4-X3) * W
         IF ((P6-U)/(P6-P5) .GE. W)  GO TO 1400
         V = F4 + (U-F5)/(X4-X3)
         GO TO 1300
C
C            THE TWO TRIANGULAR REGIONS
C
9.   700 IF (U .GT. P8)  GO TO 900
         W2 = RANF(ISEED)
         IF (W2 .GT. W)  W = W2
         IF (U .GT. P7)  GO TO 800
         X = X1 + (X2-X1)*W
         V = F1 + 2.*W*(U-P6)/(X2-X1)
         IF (V .LE. F2*W)  GO TO 1400
         GO TO 1300
10.  800 X = X5 - W*(X5-X4)
         V = F5 + 2.*W*(U-P7)/(X5-X4)
         GO TO 1300
C
C            THE TWO EXPONENTIAL REGIONS
C
11.  900 IF (U .GT. P9)  GO TO 1000
         U = (P9-U)/(P9-P8)
         X = X1 - ALOG(U)/XLL
         IF (X .LE. 0.)  GO TO 100
         IF (W .LT. (XLL*(X1-X)+1.)/U)  GO TO 1400
         V = W*F1*U
         GO TO 1300
12. 1000 U = (P10-U)/(P10-P9)
         X = X5 - ALOG(U)/XLR
         IF (W .LT. (XLR*(X5-X) +1.)/U)  GO TO 1400
         V = W*F5*U
C
C            PERFORM THE STANDARD REJECTION
C
13. 1300 IF (ALOG(V) .GT. X3*ALOG(X/X3) + X3 - X)  GO TO 100
    1400 X = BETA*X
         RETURN
         END
```

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: OREM 78009	2. GOVT ACCESSION NO. / 3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): SQUEEZE METHODS FOR GAMMA VARIATE GENERATION	5. TYPE OF REPORT & PERIOD COVERED: Technical Report
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): Bruce W. Schmeiser	8. CONTRACT OR GRANT NUMBER(s): ONR N00014-77-C-0425
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Southern Methodist University, Dallas, TX 75275	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: NR 277-236
11. CONTROLLING OFFICE NAME AND ADDRESS: Naval Analysis Program, Office of Naval Research, Arlington, VA 22217	12. REPORT DATE: August 1978
	13. NUMBER OF PAGES: 23
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	15. SECURITY CLASS. (of this report): Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Distribution of this document is unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Gamma Distribution	Rejection Methods
Simulation	Monte Carlo
Random Number Generation	Distribution Sampling

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Two algorithms are given for generating gamma distributed random variables. The algorithms, which are valid when the shape parameter is greater than one, use a uniform majorizing function for the body of the distribution and exponential majorizing functions for the tails. The algorithms are self-contained, requiring only U(0,1) variates. Comparisons are made to three competitive algorithms in terms of marginal generation times, initialization time, and memory requirements. Both algorithms are faster than existing methods, for all values of the shape parameter.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

S/N 0102-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)